(12) UK Patent Application (19) GB (11) 2 225 881 (13) A

(43) Date of A publication 13.06.1990

(21) Application No 8828487.2

(22) Date of filing 06.12.1988

(51) INT CL[5]
G06F 13/28

(52) UK CL (Edition K)
G4A AFGL

(56) Documents cited
GB 2171823 A EP 0174446 A2 WO 88/08564 A1

(58) Field of search
UK CL (Edition J) G4A AFGDC AFGDR AFGL AFI
INT CL[4] G06F
Online database: WPI

(71) Applicant
Flare Technology Limited

(Incorporated in the United Kingdom)

Unit 0, 347 Cherry Hinton Road, Cambridge, CB1 4DH, United Kingdom

(72) Inventor
Ben Cheese

(74) Agent and/or Address for Service
Flare Technology Limited
Unit 0, 347 Cherry Hinton Road, Cambridge, CB1 4DH, United Kingdom

(54) **Co-processor intrude mechanism**

(57) A simple but fast co-processor within a multi processor computing system can transfer data to and from a host processor by using a dedicated instruction in conjunction with registers and a state machine. Thus information bandwidth can be defined according to application and loss of processing power due to communication can be minimised.

GB 2 225 881 A

CO-PROCESSOR INTRUDE MECHANISM

This invention relates to a means by which two processors within a computer system may pass information between each other.

Background

Within a computing device it is often desirable to install co-processor(s) to perform tasks to which a general-purpose central processor (the CPU) is not well suited. With this arrangement it is necessary to organise the passage of data between the two processors (the co-processor and the CPU) such that the data transaction causes the minimum of disruption to the execution of programs by either processor. This is especially important where the speeds of these processors are significantly different such as in the case where the CPU is proprietary micro-processor and the co-processor is of a very simple architecture but very fast.

Essential technical features.

According to this invention there is a means built into the design of a co-processor by which the CPU may pass information to the co-processor under control of the program executing in the CPU but at times dictated by the program executing in the co-processor. With this means it is possible for the CPU to transfer data to and from the co-processor at times which are appropriate to both processors but without the loss in processing power usually associated with interrupt mechanisms or by bus sharing.

The co-processor has a special instruction (called INTRUDE) which makes the memory space within the co-processor available to the intrude registers.

The intrude registers are means by which the data is always available to the CPU. All that is required of the CPU is that when data is to be passed from the co-processor to the CPU that an extra (dummy) read is performed to inform the co-processor's intrude mechanism of the address within the co-processor from which the mechanism is to fetch the data. Data and address information are stored in the intrude registers. A state machine provides the means by which the transfer of data to and from the co-processor's memory is controlled automatically, needing no intervention from the CPU program.

Because the time at which the information is passed to the co-processor is defined by INTRUDE instructions within the co-processor's program it becomes possible for the programmer to decide at what rate he wishes information to be transferred between the two processors. This way the amount of information transfer (or bandwidth) may be decided according to the application to which the system is put.

Because the time at which the data arrives from the CPU is defined by INTRUDE instructions it becomes possible for the programmer to allow data within the co-processor to be updated in

arbritrarily sized packets. This is useful where the co-processor is executing a program which implements a digital filter (or similar digital control system) where it is desirable that coefficients of a polynomial are all updated within the same program loop at a particular time in that loop. If this is achieved then the control system can remain stable whilst its characteristics are being changed by the CPU.

Examples:

1. Reading data from Co-processor.

1.1 When the CPU needs to read data from the co-processor it first transfers the address within the co-processor (from which it requires data) to a register called the 'INTRUDE ADDRESS REGISTER'. This can either be done via an IO instruction or if the co-processor memory is also within the CPU's address space, by latching the CPU's address bus during a memory read operation from the co-processor memory.

1.2 When the co-processor encounters an intrude instruction in the instruction pipeline the address generator will take the intrude address register as the address for the following instruction cycle.

1.3 The following instruction cycle will then transfer data from the address defined by the intrude address register and place it in the 'INTRUDE DATA REGISTER'.

1.4 The CPU then will read the contents of the intrude data register.

Because the co-processor has a very simple architecture its instruction rate is very much greater than that of a proprietary micro-processor. If sufficient INTRUDE instructions are inserted into the co-processor program cycle then steps 1.2 and 1.3 will have been completed before the CPU is able to perform the final step 1.4 therefore neither processor is held up in the passage of data from one to the other.

2. Writing data from the CPU to the co-processor.

2.1 When the CPU needs to write data to the co-processor it first transfers the address within the co-processor (to which it will send data) to a register called the 'INTRUDE ADDRESS REGISTER'. This can either be done via an IO instruction or by taking the address from when the address becomes valid during a memory write operation.

2.2 The co-processor will immediately cease any further writes to the INTRUDE DATA REGISTER. This has now become the property of the CPU.

2.3 When the co-processor encounters an intrude instruction in the instruction pipeline the address generator will take the intrude address register as the address for the following instruction cycle.

2.4 The following instruction cycle will then transfer data from the INTRUDE DATA REGISTER and place it in the co-processor data location described by the intrude address register.

2.5 Once the transfer has been completed the mechanism will continue to transfer data from the the co-processor data location to the intrude data register. This has the useful property that the CPU can verify the transfer or, alternatively, monitor the progress of the data as the co-processor program updates it.

Claims

1 An Co-Processor Intrude mechanism where a co-processor in a multi-processor computing system has means for defining program time slots which are accessible to a central processor by a uniquely defined 'INTRUDE' instruction.

2 An Co-Processor Intrude mechanism as claimed in Claim 1 where means are provided for storing data and address information from the central processor until the next available time slot defined by the intrude instruction.

Published 1990 at The Patent Office, State House, 66 71 High Holborn, London WC1R 4TP. Further copies may be obtained from The Patent Office Sales Branch, St Mary Cray, Orpington, Kent BR5 3RD. Printed by Multiplex techniques ltd St Mary Cray, Kent Con. 1 87